負荷均一化に対するカクタス上の効率的なメッセージ伝搬 (最適化の数理とアルゴリズム)

AUTHOR(S):

林, 幸雄

CITATION:

林, 幸雄. 負荷均一化に対するカクタス上の効率的なメッセージ伝搬 (最適化の数理とアルゴリズム). 数理解析研究所講究録 2002, 1297: 96-106

ISSUE DATE:

2002-12

URL:

http://hdl.handle.net/2433/42657

数理解析研究所講究録 1297 巻 2002 年 96-106

負荷均一化に対するカクタス上の効率的なメッセージ伝搬

北陸先端科学技術大学院大学 林 幸雄 (Yukio HAYASHI)
Japan Advanced Institute of Science and Technology

概要: 広域ネットワークに分散配置されたサーバの負荷均一化を考え, 負荷移動のボトルネック部分をバイパスさせたカクタス上の分散アルゴリズムを提案する. この方法では, 全域木上の効率的なメッセージ伝搬法をもとに, カクタスにおける独立な閉路ごとに最適フローを変分/摂動的に求めることができる. このような初期負荷配分に依存して適応的に生成された結合構造は, 分散システムの動的な環境変化にも追従でき, 実際上有効と考えられる.

キーワード: 拡散法, 2 次計画問題, 分散アルゴリズム, Bethe 近似

1 はじめに

インターネットの急速な進展普及に伴い, 広域ネットワーク上の分散コンピューティングが近年注目を集めている [13][16]. その重要課題の 1 つとして, 分散配置された複数のサーバ (一般にコンピュータ) 間の局所的な通信と負荷移動によって, アイドリングや過度な負荷状態がなく全体が効率的に稼働するための, 負荷均一化が検討されている. これまで, 並列計算機や分散システムにおいて, 出来るだけ少ない通信回数や通信量で, 出来るだけ素早く負荷の均一化が行えるよう, さまざまなアルゴリズムが提案されている.

例えば, ある閾値より負荷量が少なくなった時点で自分の近傍に催促する Receiver Initiation や, 逆に閾値より多くなった時点で負荷の移動先を探す Sender Initiation が知られている. これらは, 処理が単純であることから用いられているが, 閾値の設定が難しいのに加えて, (大域的な均一化が保証されず) 場当たり的である [1][14]. 予め負荷移動先の順番を定めた Round-Robin 法やランダムに移動先を選択する方法なども, 均一化の精度よりも処理の単純さを重視したものである.

一方, 無駄な負荷移動をさけるため, 移動すべき負荷量の計算と, 負荷の移動という 2 ステップに基づく方法が考えられている. その最も良く知られた手法のカテゴリとして, 次元交換 (DE: Dimension Exchange) 法と拡散 (DF: Diffusion) 法がある [18]. DE 法は, 規則的な結合構造を持つ超並列計算機向きで, 一時刻に単一の通信ポートを介して一斉に右, 左, 上, 下へといった具合いに交互に近傍のプロセッサと均一化をはかる手法である. これに対して DF 法は, 非同期, マルチポート向きで, 局所的な拡散による反復計算によって各辺 (通信路) 上の負荷移動量を求めた後に, 実際の負荷移動を行う. ゆえに, 並列計算機ではなく一般の分散システムを対象とした本論文では DF 法に着目する.

DF 法は本質的には拡散方程式を反復的に解いて負荷移動量を求める手法であるが, 単純な時間差分 (5) による方法では収束が遅いため, 高速化のための種々のスキームが考えられている. 特に最近, 一般の結合構造に対して多項式を用いた OPS(Optimal Polynomial Scheme) と呼ばれる手法が提案された [5]. この手法では, 拡散方程式における離散ラプラシアン (2) の固

有値でパラメータを設定すると, 有限（相異なる固有値の数 −1）回の反復で収束することが保証される. また, 従来の FOS(First Order Scheme), SOS(Second Order Scheme), Chebyshev Scheme などを特殊形として含んだ一般化された手法である [5][7] とともに, 並列計算機に適したグラフのデカルト積の構造にも適用できる [6].

しかしながら, OPS では**予め固有値を計算しておく必要**があり, 結合構造が固定な並列計算機には適用できるとしても, 動的に環境変化が起こり得る分散システムには適してないこと, さらに, 例え各辺上の負荷移動量が求まったとしても, 移動可能な負荷が各サーバ上に存在しなければ実行できないので, （特に複数の閉路が影響し合う場合など）**負荷移動の順序が問題**となる [5].

いずれにしても, 負荷量が均衡するまでの拡散の収束速度には離散ラプラシアンの固有値, すなわち, 結合構造と重み値（グラフの半径やボトルネック部分など）が関係しており, トポロジーによる比較も検討されている [4]. 但し, 限られた辺数で最も速い拡散伝搬を実現する結合構造をみつけることは, 正則グラフの場合でも非常に難しい [15].

本論文では, 従来の OPS の問題点をふまえた上で, 広域ネットワーク上に分散配置されたサーバ（一般にコンピュータ）の負荷を均一化する手法を検討する. 従って, この問題は, WWW サーバへのアクセスやストリーミングメディアコンテンツに対する, 末端のスイッチにおけるローカルな負荷分散 [1] とは異なることに注意されたい.

具体的には, **閉路を持たない全域木上で効率的に負荷移動量を求めた後, 負荷移動のボトルネックとなる辺のフローをカクタスによって適応的にバイパスする（移動条件を緩和する）手法**を提案する. 最適なトポロジーとは限らないが, 最低限の連結性を保証した全域木から, 高々 1 個の頂点のみを共有する独立した閉路で構成されるカクタスを初期負荷配分に対して適応的に生成する, このような結合構造は実際上有効と考えられる.

2 分散システムの負荷均一化

2.1 問題設定

まず, 並列計算機と比較した分散システムの特徴：緩やかな結合, プロセスの独立性, 異質性 [16] を考慮して, 以下の設定を考える. 但し, 議論の単純化のため, 本論文ではサーバ性能は全て同一とする.

- 先行研究 [4][5][6][18] と同様に, 負荷量は任意に分割可能とする. 例えば, WWW クローラ [12] が探索すべき多量の URL 数や, 科学技術計算における同一問題に対する膨大なパラメータの組み合わせ [13] など, プロセス間の順序や依存関係を考えなくて良い課題を主な対象とする.

- サーバ間の結合は, インターネット上の IP パケットルーティングに基づく論理的なものを想定し, 通信チャネルも仮想的にマルチポートとみなす. 一般性を考えて広域ネットワークを想定しているが, LAN 内の PC クラスタなどもこれに含まれる.

- 結合構造は規則的である必要はなく, すなわち, 各頂点の次数（辺の数）が違っても良い疎密な結合部分が混在した, 一般の異質なトポロジーを考える.

- 地理的に離れた箇所との結合もあり得るので, 全体の同期化は困難と考えられることから, 局所的な非同期処理を前提とする.

このように本論文では, 負荷指標は処理すべき量（データ量やプロセス数などに相当）と定義する. また, インターネット上では通信経路が動的でデータ転送速度や遅延などが不確定なため, こうした通信効率に関する辺の重みは考えないことにする. 少なくとも, 移動量の計算では, 負荷量に関するわずかな数値データを送受するだけなので, 遅延等は無視できる. その際, 通信より処理の方が支配的と考えて差し支えない. 但し, 全く別問題なので本論文では議論しないが, 負荷の定義や次章以降で述べる拡散モデルとの対応付けを再考修正すれば, トラフィックの負荷分散などにも適用できるかも知れない.

一般に, 最小化すべきサーバのレスポンス時間は, 通信頻度や CPU 使用率さらにプロセス粒度などにも依存するため, その推定は困難であり, 通信と処理（CPU やディスク, I/O アクセスなど）をどのように切り分けて負荷を定義すべきかについて未だ確立されたものはない [18].

2.2 拡散法の 2 次計画問題への帰着

自己ループや多重辺のない無向グラフ (V, E) に対する離散ラプラシアン [2][17] を考える. その行列ベクトル表現は,

$$\mathbf{Lf} = \begin{bmatrix} \cdots & \cdots & -w_e & \cdots & \cdots \\ \vdots & \ddots & 0 & \vdots & \vdots \\ -w_e & 0 & \sum w_e & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{pmatrix} f(1) \\ \vdots \\ f(u) \\ \vdots \\ f(m) \end{pmatrix}, \tag{1}$$

その $u \in V$ 成分は,

$$Lf(u) = -\sum_{v \sim u} w_e (f(v) - f(u)), \tag{2}$$

と表される. ここで, $m = |V|$, $n = |E|$, $f(u)$ は各頂点 $u \in V$ の負荷量を表し, $v \sim u$ は頂点 u の隣接点 v の集合（$\sum_{v \sim u}$ は u の隣接点についての和をとること）を表す. また, 辺 $e = (u, v)$, $w_e > 0$ は各辺 $e \in E$ の重み： DF 法では数値計算の加速パラメータともみなされるが, 本論文では, サーバ間の結合の安定度と解釈する（ping などの通信時間の変動が少ないもの程, 負荷移動を多めに行うように重み付ける）.

さて, 拡散方程式

$$\frac{\partial \mathbf{f}}{\partial t} = -L\mathbf{f}, \tag{3}$$

を考えよう. (3) の時間発展は, 総負荷量 $\sum_{u \in V} f(u)$ を保存し, 負荷均一解

$$\bar{f} \stackrel{\text{def}}{=} \frac{\sum_{u \in V} f(u)}{|V|}, \tag{4}$$

との自乗誤差 $\sum_{u \in V} (f(u) - \bar{f})^2$ を単調減少させながら収束することが知られている.

その時間差分版は,

$$\mathbf{f}^k = (I - \Delta t L)\mathbf{f}^{k-1} = \underbrace{F \times \ldots \times F}_{k}\mathbf{f}^0, \tag{5}$$

である. ここで, $F \stackrel{\text{def}}{=} I - \Delta t L$, $\mathbf{f}^k$ は k 回目の反復における負荷量ベクトル, $\mathbf{f}^0$ は初期負荷量ベクトルを表す. また, Δ は時間の刻み幅（DF 法のパラメータに相当）で, $1 \leq \Delta t \times \sum_{e \in E_u} w_e$, E_u は u の接続辺の集合とする [10]. 以後, $\Delta t \times w_e$ を改めて w_e で表記する（$\Delta t = 1$ に相当）.

負荷均一化の DF 法としては, 式 (3) の右辺すなわち式 (2) によって, 各頂点 u のサーバが近傍 v との負荷量の差に従って移動すべき量を求める点が本質的である. 但し, 具体的な計算手順においては, 先に述べたようにさまざまな高速化スキームが施されている.

一方, 式 (5) は,

$$\begin{aligned} y_e^{k-1} &= -w_e(f^{k-1}(v) - f^{k-1}(u)), \\ z_e^k &= z_e^{k-1} + y_e^{k-1}, \quad z_e^0 = 0, \\ f^k(u) &= f^{k-1}(u) - \sum_{e \in E_u} y_e^{k-1}, \end{aligned} \tag{6}$$

と等価になる. さらに, 拡散方程式 (3) あるいは (5) は, 以下の 2 次計画問題と等価になる [5][10].

$$\min \quad \tfrac{1}{2}\mathbf{z}^T W^{-1}\mathbf{z}, \tag{7}$$

$$s.t. \quad B\mathbf{z} = \mathbf{f}^0 - \bar{\mathbf{f}}, \tag{8}$$

ここで, $W = diag(w_e)$, B: 接続行列, z_e: 辺 $e \in E$ 上の負荷移動フロー, $\bar{\mathbf{f}} = (\bar{f}, \ldots, \bar{f})$: 負荷均一解ベクトルとする.

その等価性は以下のようにして確かめることができる [10]. 行列ベクトル表記で (6) は $\mathbf{y}^k = WB^T\mathbf{f}^k$ と書け,

$$\mathbf{z}^k = \sum_{l=0}^{k}\mathbf{y}^l = WB^T\sum_{l=0}^{k}\mathbf{f}^l = WB^T\left(\sum_{l=0}^{k}(I - L)^l\mathbf{f}^0\right),$$

となる. ここで, (1) の L が対称行列であることから, その固有値 $0 = \lambda_1 < \lambda_2 < \ldots, \lambda_K < 2$ と, 1 次独立な固有ベクトル $\mathbf{u}_j$, $(j = 1, \ldots, K)$ を用いて, a_j を係数とした $\mathbf{f}^0 = \sum_{j=1}^{K} a_j\mathbf{u}_j$ と表現できる.

これより,

$$\sum_{l=0}^{k}\mathbf{y}^l = WB^T\left(\sum_{l=0}^{k}\sum_{j=1}^{K}(1 - \lambda_j)^l a_j\mathbf{u}_j\right),$$

となる. $a_1 = \bar{f}$, $\mathbf{u}_1 = (1, \ldots, 1)^T$ による $B^T\mathbf{u}_1 = 0$ と, $\sum_{l=0}^{\infty}(1 - \lambda_j)^l = \frac{1}{\lambda_j}$, $(j \neq 1)$ より,

$$\mathbf{z} = \sum_{l=0}^{\infty}\mathbf{y}^l = WB^T\mathbf{d}, \quad \mathbf{d} = \sum_{j=2}^{K}\frac{a_j}{\lambda_j}\mathbf{u}_j,$$

を得る.

この $\mathbf{z}$ が条件 (8) を満足することは,

$$\begin{aligned} B\mathbf{z} &= BWB^T\mathbf{d} &&= L\mathbf{d} \\ &= \textstyle\sum_{j=2}^{K} a_j\mathbf{u}_j &&= \mathbf{f}^0 - \bar{\mathbf{f}}, \end{aligned}$$

また, (7) の最小解であることは, 仮に $B(\mathbf{z}+\Delta\mathbf{z}) = \mathbf{f}^0 - \bar{\mathbf{f}}$, かつ, $(\mathbf{z}+\Delta\mathbf{z})^T W^{-1}(\mathbf{z}+\Delta\mathbf{z}) < \mathbf{z}^T W^{-1}\mathbf{z}$ を満たす $\Delta\mathbf{z} \neq 0$ が存在したとすると,

$$2\mathbf{z}^T W^{-1}\Delta\mathbf{z} + (\Delta\mathbf{z})^T W^{-1}\Delta\mathbf{z} < 0,$$

となるが, 上記第 1 項は

$$2\mathbf{z}^T W^{-1}\Delta\mathbf{z} = 2(WB^T\mathbf{d})^T W^{-1}\Delta\mathbf{z} = 2\mathbf{d}^T B\Delta\mathbf{z} = 0,$$

なので, 第 2 項 $(\Delta\mathbf{z})^T W^{-1}\Delta\mathbf{z} < 0$ のみが残る. しかしながら, W は正定値であることと, 仮定 $\Delta\mathbf{z} \neq 0$ より, これは非負にならず矛盾する. よって, 最小解 $\mathbf{z}$ は唯一である.

上記の 2 次計画問題の応用上の意味に注目しよう. すなわち, 負荷の均一化条件 (8) は巡回路上の無駄なフローを含むものも可能解とするが, 2 次計画問題と等価な拡散方程式 (3) は, 暗黙的に (7) による (結合の安定度に従った) 重み付きコスト最小化によってこれを防いでいることになる.

2.3 効率的な木上のメッセージ伝搬法

一方, 木の場合は巡回路自体を持たないことから, DF 法の各種のスキーム [5][6][7] のような反復計算を施すことなく, メッセージ伝搬によって直接的に各辺の負荷移動量を求めることができる. このような効率的な手法は, 負荷均一化における Tree Walking Algorithm [3][14] として知られているとともに, 統計物理やグラフィカルモデルの因果推定などの分野における Bethe 近似や Junction Tree Algorithm[11], Belief Propagation [19] などと本質的に同じものである.

< TWA: Tree Walking Algorithm >

1. 葉から根に向かって順に各頂点 u の親へ, 負荷量 $f(u)$ の累積値を送る.
2. 根に累積値が全て到達したら, 葉に向かって順に各頂点 u の子へ, 根で求めた (4) による $\bar{f}$ の値をブロードキャストする.
3. $\bar{f}$ が到達した葉から根に向かって順に, 各頂点 v が子供からの出入り量 z_e を考慮して, v から親 w への辺 e' 上のフロー (移動すべき負荷量)

$$z_{e'} = f(v) - \bar{f} + \sum_e z_e,$$

を求める (v が葉の時は $z_e = f(v) - \bar{f}$) .
4. フローの決定直後, 非同期に負荷移動ができる辺から実行する.

TWA は一見かなり集中制御的なように思われるが, 根は動的に決定することができ (累積値が最終的に到達した頂点とする) , その計算量は (4) による $\bar{f}$ だけが余分なだけで, 通信量も累積値の送受のみなので他の頂点と同程度である. 従って, 個々の負荷量などを共有する必要がなく, 局所的な通信のみで負荷移動量が直接的に求められる. もちろん, 各頂点におけるサーバは, その隣接点や結合辺のみならず, (親のみが隣接点かどうかによって) 自分が葉であるか間接点であるかを局所的に知っているものとする.

3 カクタスによる負荷移動の条件緩和

カクタスとは, 図1や4のように, 各閉路が高々1個の頂点のみを介して連結したグラフである. 本章では, 前章で述べた木上のメッセージ伝搬で求めた各辺のフローのうち, その移動量が多いボトルネックな辺のフローをバイパスさせることで, 負荷移動の条件を緩和する方法を検討する. 本論文では, IP パケット通信によるサーバ間の結合を想定しているので, 物理的なバイパスではないことに注意されたい.

バイパス辺の追加によって拡張されたネットワークに対して, (その2次計画問題の) 最適フローを求める際, 相互に影響し合う閉路ができないようにする為, カクタスを考える. なぜなら, 独立な閉路なら, それぞれのバイパス辺の最適フローが個別に求まるからである.

3.1節では, 予めカクタス構造が固定された場合を考える. 3.2節では, 初期負荷配分に応じて適応的にカクタスを生成する方法を検討する.

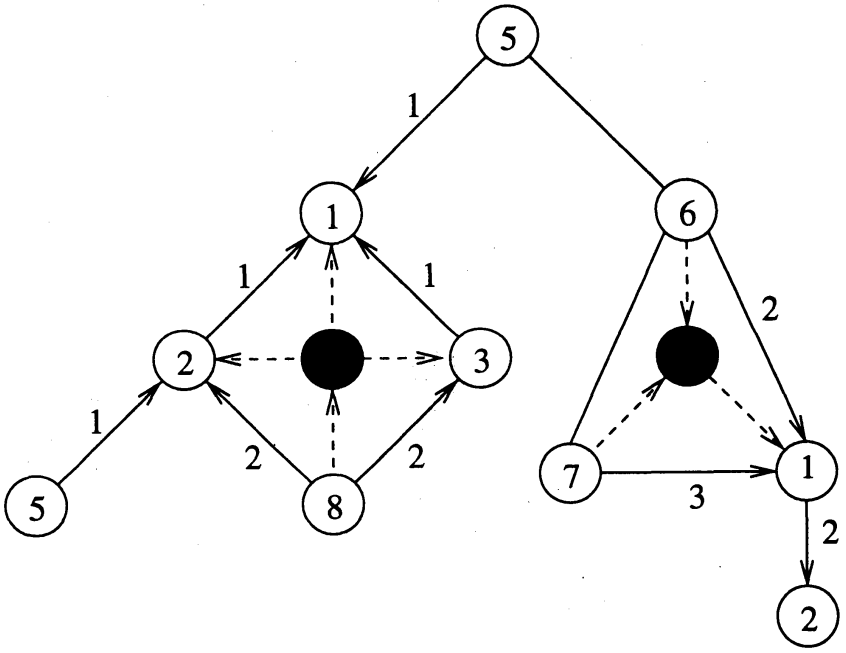

図 1: 固定されたカクタスに対する仮想木と最適フローの例

3.1 固定されたカクタスの場合

地理的な距離などでボトルネックになりそうな辺をバイパスして予めカクタス構造が固定された場合は, 図1のように仮想的な木を構成してTWAを適用する. 図1における各頂点内の数字は初期負荷量を, 各辺上の数字は以下の手順で求めた矢印方向のフローを表す (単純化のために, この例では辺の重みは全て1とした).

結果的に, 図1左側の閉路では左右2つのパスができ, どちらか片方のパスのみで負荷を移動する場合 (例えば②と⑧をつなぐ辺を除いた場合) と比べて, 負荷移動量が少なくなっていることがわかる. 従って, 移動すべき負荷の量が少なくなることで, フロー計算後の実際の負荷移動の際の条件が緩和される (負荷が貯まるまで待たなくて良い) ことになる.

< 仮想的な木上の TWA >

1. まず, 各閉路に対して ghost 頂点（図の黒丸）を考え, 閉路上の各頂点と ghost 頂点を仮想辺（図の破線）で結ぶ.
2. 各閉路上の辺を一時的にないものと考え, 仮想辺と閉路以外の辺による仮想的な木を考える.
3. 仮想木上で TWA を実行し, 各辺のフローを求める.
4. 各閉路ごとに, 仮想辺のフローを閉路上の各辺に順に割り付ける.

最後の, 仮想辺のフローの割り付けに関して補足する. まず, 閉路上のある辺の方向が 2 つの隣り合う仮想辺の方向と一致する場合（閉路上の頂点から ghost 頂点に入出, あるいは出入する方向）は, それらの仮想辺の少ない方のフローを対応する閉路上の辺に割り付ける. 一方, こうした方向の一致が成り立たない閉路上の辺は, 上記に従って割り付けた辺から右まわり（あるいは左まわり）に順に, その仮想辺に接続する頂点に関する閉路上の辺にフローを割り付ける. 但し, 1 つの仮想辺がそれを挟む 2 つの閉路上の辺に関係するときは, フローを分割する（$w_e \neq 1$ のときは重み値で比例配分）. 例えば, 図 1 の左側の閉路では, 図 2 のようにして割り付けが行われる.

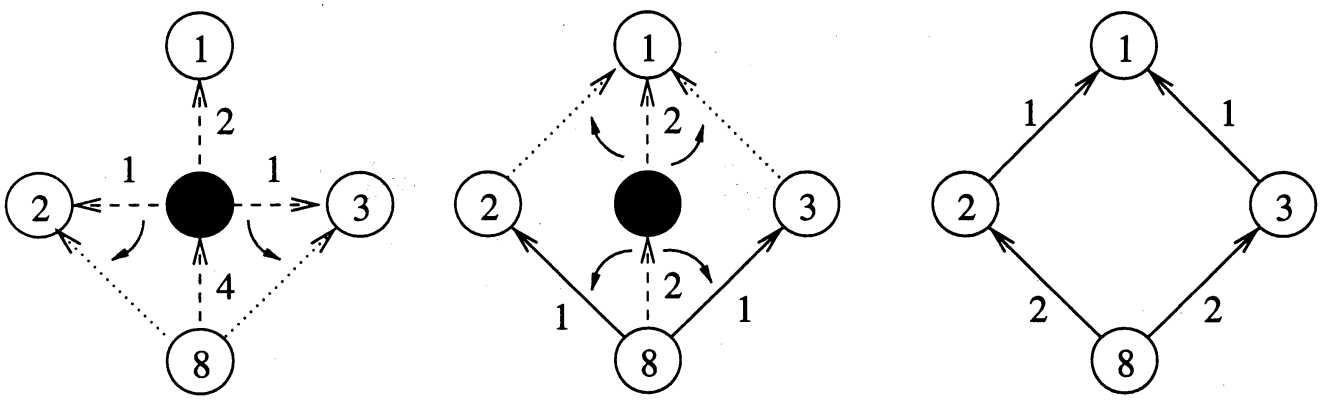

図 2: 仮想辺のフローの割り付け（左から右）

3.2 適応的なカクタスの生成

本節では, 初期負荷配分に従って全域木から適応的にカクタスを生成しながら最適フローを求める方法を, 分散アルゴリズムとして提示する.

図 3 のように, 葉でない頂点 u に関する接続辺のうち, 最大コストとなるボトルネック辺が

$$e = \arg \max_{e \in E_u} \left\{ \frac{|z_e|^2}{w_e} \right\},$$

だとする. z_e は先の全域木上の TWA で求めた辺 e のフローで, 初期負荷配分に依存した量となる.

その辺 $e = (u, v) \in E_u$ と逆方向のフローで, バイパス辺 $e'' = (w, v)$ の付加によって総コスト (7) が最も小さくなる u の接続辺 $e' = (w, u) \in E_u$ を探す. それは, バイパス辺の付加で拡張された結合構造に対する 2 次計画問題（変数なども書き直して）において,

変分/摂動的なコストの削減: $\min \ \frac{1}{2}\mathbf{z}^T W^{-1}\mathbf{z},$

バイパス化なので負荷均衡は不変: $s.t. \ B\mathbf{z} = \mathbf{f}^0 - \bar{\mathbf{f}},$

を求めていることに他ならない. ここで, ネットワークが拡張されるので変分/摂動的と呼んでいる. また, 付加したバイパス辺 e'' のフローによって, e' のフローが増加しないように逆方向のフローとなる辺を考えている.

さて, バイパス辺 e'' のフロー Δz によるコストの変化分は

$$\delta C(\Delta z) \stackrel{\text{def}}{=} \frac{(z_e - \Delta z)^2}{w_e} + \frac{(z_{e'} - \Delta z)^2}{w_{e'}} + \frac{\Delta z^2}{w_{e''}} - \left(\frac{z_e^2}{w_e} + \frac{z_{e'}^2}{w_{e'}}\right),$$

となり, 最もコストの減少が大きいときは, $\frac{\partial(\delta C)}{\partial(\Delta z)} = 0$ となることから, これを解いて具体的に

$$\Delta z_{opt} = \frac{w_{e'}w_{e''}z_e + w_e w_{e''} z_{e'}}{w_{e'}w_{e''} + w_e w_{e''} + w_e w_{e'}} > 0,$$

を得る. 逆に, これよりコストの減少 $\delta C(\Delta z_{opt}) < 0$ を示すこともできる（常にコストの減少が保証される）. 以上の手順を, 非同期で局所処理に基づく分散アルゴリズムとしてまとめる.

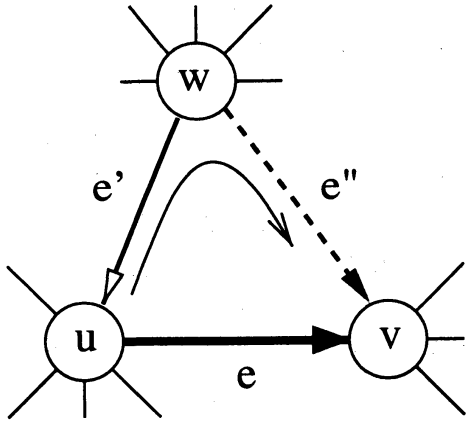

図 3: ボトルネック辺のバイパス化

< 適応的なカクタス上の負荷均一化の分散アルゴリズム >

1. 既存の分散アルゴリズムにより最小全域木を生成する [8][9]. その際, ad hoc な設定ではあるが, 地理的な距離などを考える.
2. TWA により全域木上の辺のフローを求める.
3. 葉以外の各頂点においてそれぞれ独立にバイパス辺を探す. 但し, 図 4 のように, 隣接する複数のバイパス辺の候補は, 時刻印などによる相互排除によってカクタス性（各閉路が高々1 頂点のみを共有する）を満足するように選択される.
4. カクタス（木とバイパス）上の各フロー $z_e - \Delta z_{opt}$ と Δz_{opt} の決定後, 非同期に負荷移動ができる辺から実行する.

このように, 各閉路が独立であることから局所的な処理のみによって, 最終的に生成されたカクタスに対する 2 次計画問題と等価な解が得られる.

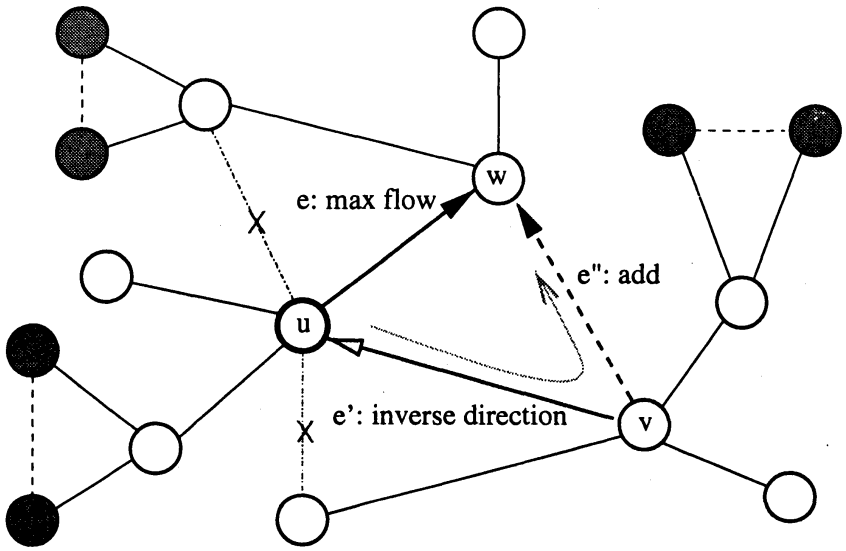

図 4: バイパス候補の相互排除

ここで, 相互排除の際, 時刻印以外に分散システムの環境変化や, コスト減少幅 $\delta C(\Delta z_{opt})$ などに従ってバイパス辺を選択しても良い. 但し, これは局所的に最適なコスト減少であって, それによって排除された他の閉路によるコスト減少分（の組み合わせ）を考慮していないので, さまざまなトポロジーを考えたときの全体のコストとしては最適かどうかはわからない. 時刻印による選択の場合も, 最適かどうかは同様にわからない. すなわち, あくまで結果的に生成された結合構造に対する最適フローが得られるだけなので, トポロジーまで変化させたときに最もコストが小さくなることを保証するものではない. しかしながら, ある程度のサーバ数であれば完全グラフが非現実的なのは明らかな一方, 限られた辺数で最も速い拡散伝搬を実現するトポロジーを見つけることは, 正則グラフの場合でも非常に難しい [15].

さて, 上記のようにカクタスを生成する際, Ternary（三角閉路）で構成するのが妥当と考えられる. その主な理由として,

- 相互排除は隣接頂点に関する交互の組合せ（図 4 では u, v, w の三角形か, × 印の辺を持つ v と w に関する三角形のどちらかを選択すること）に限定され, 最近接頂点より離れたところとは独立,
- 単なる仲介転送のみの長いパスを回避し, 隣接頂点との直接の通信のみで処理できる,
- バイパス辺の両端の頂点は地理的にも近い可能性が高い,

などがあげられる. したがって, こうして生成されたカクタスは, 最適とは限らないが, 初期負荷配分に適応的で実際に妥当なトポロジーであると考えられる.

4 おわりに

本論文では, 広域ネットワークに分散配置されたサーバの負荷均一化を考え, 従来の拡散法における固有値計算の必要性や複数の閉路上の負荷移動の順序の問題などをふまえて, カクタス上の分散アルゴリズムを提案した.

まず, 閉路を持たない全域木上の効率的なメッセージ伝搬法を, 固定されたカクタスに拡張した. 次に, 最低限の連結性を保証する全域木上の各辺のフロー（負荷移動量）に対して, ボトルネックとなる辺にバイパス辺を付加することで移動すべき負荷量を減少させる（移動条件を緩和させる）, 適応的なカクタス生成法を提示した. その際, 拡散法 (5) が 2 次計画問題 (7)(8) と等価であることを利用して, カクタスにおける独立な閉路ごとに最適なバイパスフローを変分/摂動的に求めることができる点が, 局所分散処理に適している.

このような初期負荷配分に依存して適応的に生成された結合構造は, 分散システムの動的な環境変化に追従する場合にも適し, 最適なトポロジーとは限らないが, 実際上有効と考えられる. 今後は, WWW クローラ [12] などの具体的な応用課題に対する実装等を通じて, 提案した分散アルゴリズムの性能評価についても検討していきたい.

謝辞： 本研究における辺の重みの定義に関して有益な御指摘を頂きました, 京都大学大学院情報学研究科の茨木 俊秀 教授, ならびに, 文献 [11] を御教え頂きました, 東京工業大学大学院総合理工学研究科の樺島 祥介 助教授に感謝申し上げます. 本研究の一部は, 文部科学省科学研究費 13680404 の援助を受けている.

参考文献

[1] T. Bourke. *Server Load Balancing*, O'Reilly, 2001.

[2] F.R.K. Chung. *Spectral Graph Theory*, Chapter 1, Amer. Math. Soc., 1994.

[3] S.K. Das, D.J. Harvey, and R. Biswas. "Adaptive Load-Balancing Algorithms Using Symmetric Broadcast Networks," *Journal of Parallel and Distributed Computing*, vol. 62, pp. 1042-1068, 2002.

[4] T. Decker, B. Monien, and R. Preis. "Towards Optimal Load Balancing Topologies," A. Bode et al. (Eds): Euro-Par2000, *LNCS* 1900, pp. 277-287, 2000.

[5] R. Diekmann, A. Frommer, and B. Monien. "Efficient Schemes for Nearest Neighbor Load Balancing," *Parallel Computing*, Vol. 25, pp. 789-812, 1999.

[6] R. Elsässer, A. Frommer, B. Monien, and R. Preis. "Optimal and Alternating-Direction Load Balancing Schemes," In P. Amestoy et al. (Eds.): Euro-Par'99, *LNCS*, 1685, pp. 280-290, 1999.

[7] R. Elsässer, B. Monien, and R. Preis. "Diffusive Load Balancing Schemes on Heterogeneous Networks," *Proc. of SPAA*, pp. 30-38, 2000.

[8] R. Gallager, P. Humblet, P. Spira. "A Distributed Algorithm for Minimun Weight Spanning Trees," *ACM Trans. on Prog. Lang. and Systems*, Vol. 5, No. 1, pp. 66-77, 1983.

[9] L. Higham, Z. Liang. "Self-Stabilizing Minimum Spanning Tree Construction on Message-Passing Networks," In J. Welch (Eds.): DISC 2001, *LNCS*, 2180, pp. 194-208, 2001.

[10] Y.F. Hu, R.J. Blake. "An Improved Diffusion Algorithm for Dynamic Load Balancing," *Parallel Computing*, Vol. 25, pp. 417-444, 1999.

[11] 樺島 祥介. "グラフィカルモデルと平均場近似," **信学技報**, NC2001-112, pp. 39-46, 2002.

[12] 森 英雄, 河野 浩之. "実測データに基づく分散協調型 WWW データ収集アルゴリズムの性能評価," **第 2 回インターネットテクノロジーワークショップ**, 1999.

[13] 佐藤 三久. "グローバルコンピューティングへの期待 [8]," *Computer Today*, No. 104, 2001.

[14] W. Shu, and M.Y. Wu. "Runtime Incremental Parallel Scheduling on Distributed Memory Computers," *IEEE Trans. on Parallel and Distributed Sysytems*, vol. 7, no. 6, pp. 637-649, 1996.

[15] 砂田 利一. "**離散スペクトル幾何学**," 上野 他編, **数学のたのしみ**, No.12, pp. 67-80, 日本評論社, 1999.

[16] V.S. Sunderam, and G.A. Geist. "Heterogeneous Parallel and Distributed Computing," *Parallel Computing*, Vol. 25, pp. 1699-1721, 1999.

[17] 浦川 肇. **ラプラス作用素とネットワーク**, **第 7 章**, **裳華房**, 1996.

[18] C. Xu, and F.C.M. Lau. Load Balancing in Parallel Computers -Theory and Practice-, Kluwer Academic Publishers, 1997.

[19] J.S. Yedida, W.T. Freeman, and Y. Weiss: "Bethe free energy, Kikuchi approximation, and belief propagation algorithm," http://www.merl.com/papers/TR-2000-26, 2000.